

**Attorney Docket No. NAT199-143085****REMARKS**

Claims 1-12 are pending in the application. Claims 13-62 were previously withdrawn without traverse in response to the Examiner's restriction requirement. Claims 1-12 have been rejected by the Examiner.

Applicant thanks the Examiner for the in-person conference held on September 14, 2004. The Examiner's comments and questions were quite helpful and it is hoped that the explanations provided by the inventor/applicant were instructive.

1. **Amendments to Specification and Claims**

The changes to the specification and claims above are for clarification to eliminate potentially confusing use of the synonymous terms "group proxy shell" and "group logic shell" so that only the term "group logic shell" is used. The meaning and scope of the specification and claims is not intended to be modified by this change. Support for this change can be found by frequent use of the term "group logic shell" throughout the specification and in the figures (see for example Figures 2 through 9 in the "Key" and page 16, line 10 of the specification).

The abstract has been amended so that it is less than 150 words in response to the objection raised by the Examiner in paragraph 6 of the office action.

2. **103(a) Rejection over U.S. Patent No. 5,903,725 to Colyer (Paragraph 8 of Office Action)**

Claims 1-12 have been rejected by the Examiner over U.S. Patent No. 5,903,725 to Colyer. Applicant respectfully disagrees.

The Examiner's rejection of these claims in view of Colyer is based upon the assumption that the proxy and alternative proxies disclosed in Colyer are analogous to the group proxy shell and service proxy in the present application. This is not the case. The Examiner may have been

**Attorney Docket No. SYN006-05**

confused by our interchangeable use of terms "group logic shell" and "group proxy shell". The use of "group logic shell" is more appropriate and the claims and specification have been amended to reflect this clarification.

Importantly, in the present invention the group proxy is comprised of a group logic shell and one or more service proxies. A service proxy (or alternatively a "member service proxy") is the means of communicating with a remote service; it is these remote services that form the group. The transition is not merely swapping group proxies (as in Coyler where a first proxy is swapped with a second proxy); in the present invention the transition is implemented by updating the group proxy with the appropriate group logic shell and the relevant member service proxies. Equally important, the group logic shell contains the logic that implements the group abstract data type: group communication, member failure detection, handling and masking, consensus algorithms, leader election algorithms, and so forth. In short, the group logic shell contains much of the results from algorithmic research in distributed computing. It is far more complex than a simple fail-over mechanism.

This is explained in detail at page 16, line 10 through page 17, line 23 of the disclosure which is reproduced below for the Examiner's convenience with emphasis added:

The group proxy 40, 42 is made up of a group logic shell 30, 32 and one or more service proxies 10a, 12a, 14a, 16a, 18a. The group logic shell 30, 32 contains all of the necessary group logic for a client to interact with a group of services. Assuming there is a defined interface (e.g. syntax) to call a service, the group logic shell 30, 32 contains this interface to present to clients 2, 4. The group logic shell 30, 32 contains the logic to intercept client 2, 4 commands to a group 50, 52, store them, and retransmit the commands at a later time. The group logic shell 30, 32 may also contain logic to copy or redirect client 2, 4 communication to other services. However, the group logic shell 30, 32 does not contain the necessary mechanisms, such as wire protocol implementation, to

## Attorney Docket No. SYN006-05

interact with the services 10, 12, 14, 16, 18 within a group. These are contained within the service proxies 10a, 12a, 14a, 16a, 18a. The group service 24 bundles the group logic shell 30, 32 with one or more service proxies 10a, 12a, 14a, 16a, 18a to form a group proxy 40, 42.

As shown in Figure 2, there are separate group logic shells for a CC group 30 and for peer group 32. In fact, in the current embodiment there are two group logic shells for each group, one peer and one CC.

...

The use of a group logic shell to form a group proxy is an improvement of the current invention. It makes it possible to create and reconfigure group proxies on the fly as the application is running. It enables an architecture where, in most cases, only service proxies in the group proxy need to be updated as services are added and deleted from a group, instead of replacing the entire group proxy. Alternatively, logic shells may be changed, perhaps to switch between peer and CC modes, without replacing the service proxies.

Similarly, the definition of a "service proxy" is provided in the paragraph beginning on page 1, line 23 of the disclosure which is reproduced below for the Examiner's convenience with emphasis added:

Figure 1 shows a typical distributed application of the existing art. There are two clients 2, 4 and four services 10, 12, 14, 16 that the clients 2, 4 might need. Each service has a service proxy 10a, 12a, 14a, 16a which is a module of mobile code that can be used by clients to invoke that service. A service proxy 10a, 12a, 14a, 16a contains the code needed by a client 2, 4 to interact with a service. For instance if a service is a digital camera on a robotic arm, the interfaces might include Initialize(), Zoom(), Rotate() and Get\_Picture(). The service proxy 10a, 12a, 14a, 16a may also provide the expected return values for the service, which might include error codes as well.

**Attorney Docket No. SYN006-05**

Thus, in a system made up by clients and grouped services, the group logic shell contains the logic that enables the client to talk to the group, and the service proxy contains the logic to talk to a particular service within a group. Service proxies are well known in the prior art, as used by Colyer and many others. Further, Colyer does not teach the grouping of services together, or the dynamic creation of a group proxy comprised of a group logic shell and a service proxy. Since member service proxies are mobile code, these can be dynamically inserted in the group logic shell when a new member joins. That is, mobile member service proxy code means that the group proxy can be updated as necessary, and not only in accordance with a priori settings.

In addition, as set forth above, the service proxy of the present invention necessarily is made of mobile code (see page 2 lines 4 through 18 for an explanation of mobile code). The proxy and alternative proxy of Coyle are not comprised of mobile code and do not contain any group logic.

Not only does Colyer not disclose mobile code, it makes no mention of the grouping of services. Therefore it would not be obvious to apply or extend any of the techniques of Colyer to grouped services.

With respect to claim 2, Colyer does not disclose any transition in groups because it does not disclose any groups. Further, Colyer does not disclose (i) the addition of a new service to a group, (ii) the removal of a service from a group, or (iii) a change in group mode (for example from coordinator cohort to peer).

With respect to claim 4, Colyer does not distinguish between the group proxy, a service proxy and a group logic shell, but merely switches between the proxies.

**Attorney Docket No. SYN006-05**

With respect to claim 5, the cited section of the Colyer reference does not appear to disclose a coordinator cohort group or a peer group.

With respect to claim 6, Colyer does not disclose any use of a lookup service.

With respect to claims 7-12, Applicant relies on the arguments set forth with respect to claims 1-6 above.

\*\*\*\*\*

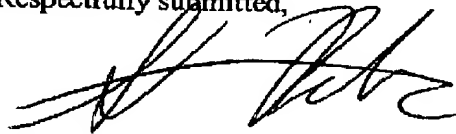
It is respectfully submitted that all claims in the application are allowable.

Reconsideration and withdrawal of all rejections are respectfully requested. Favorable notice to this effect and early Notice of Allowance are earnestly solicited.

Should the examiner have any questions and in order to expedite prosecution of this Application, the Examiner is encouraged to contact the undersigned directly.

**Please note the new contact information below for Attorney for the Applicant.** A Notice of Change of Correspondence Address was previously submitted on May 24, 2004.

Respectfully submitted,



Date: September 21, 2004

Stuart D. Rudoler, Reg. No. 49,059  
Attorney for Applicant  
Stuart Rudoler LLC  
Suite 300  
2 Bala Plaza  
Bala Cynwyd, PA 19004  
Tel: 610-660-7753  
Fax: 267-200-0796